DE SYSML A MODELICA : AIDE A LA FORMALISATION DE MODELES DE SIMULATION EN CONCEPTION PRELIMINAIRE

Roland Renier (1), Raphaël Chenouard (1)

(1)ECN-IRCCyN, 1 rue de la Noë 44000 Nantes
Roland.Renier@irccyn.ec-nantes.fr, Raphael.Chenouard@irccyn.ec-nantes.fr

Résumé:

Cet article s'insère dans un contexte de conception préliminaire d'un système complexe dans lequel l'approche de l'ingénierie système basée sur les modèles est utilisée. SysML et Modelica sont les deux langages de modélisation sur lesquels nous nous focaliserons. Ces deux langages possèdent des concepts ou des approches de modélisation similaires et bien qu'ils n'aient pas la même finalité, ils sont tous les deux utilisés pour modéliser le comportement d'un système. SysML permet une modélisation graphique et visuelle alors que Modelica est un langage textuel destiné à la simulation des modèles. Dans le but d'aider à la formalisation de modèles durant la phase de conception préliminaire, des analogies entre les langages SysML et Modelica sont présentées. Cet article reprend les grands concepts de ces 2 langages et leurs relations. Un cas d'application de conception haut niveau de la partie propulsive d'un navire est présenté.

Mots clés: modélisation, conception préliminaire, ingénierie système.

1 Introduction

Dans cet article, nous nous intéressons au contexte de l'aide à la décision en conception préliminaire et plus précisément lors du dimensionnement du système [1]. Nous suivons alors l'approche du Model-Based System Engineering (MBSE), qui consiste à se baser sur des modèles pour assurer toute activité autour d'un système, ici sa conception. Se baser sur des modèles permet de représenter la connaissance à l'aide d'un formalisme bien défini. Ainsi, cette approche permet d'améliorer la réutilisabilité et la traçabilité des connaissances formalisées.

De plus, l'ingénierie système permet d'analyser les différentes facettes d'un système : ses fonctions, sa structure et son comportement [2]. En conception préliminaire, l'objectif est de déterminer les performances du système par rapport à des choix de concepts et de dimensions des composants retenus. Il est alors nécessaire de simuler ce système pour des situations de vie données et obtenir ses performances pour les confronter aux exigences du cahier des charges.

L'objectif de cet article est d'aider à la formalisation de modèles de simulation. Pour notre étude, nous avons choisi les langages SysML (Systems Modeling Language) et Modelica. Nous présentons les analogies existantes entre ces deux langages ayant des finalités différentes. SysML est un langage de modélisation graphique utilisé pour la spécification conceptuelle de systèmes, alors que Modelica a pour vocation de simuler des systèmes formulés mathématiquement. L'utilisation de SysML dans cet article est restreinte à deux diagrammes qui permettent de définir la structure du système : le Block Definition Diagram (BDD) et l'Internal Block Diagram (IBD). Les concepts inclus dans ces diagrammes sont proches de ceux qui permettent la spécification structurelle d'un système en Modelica.

La section 2 présente le contexte de l'ingénierie système et les deux langages sur lesquels nous basons notre travail : SysML et Modelica. La section 3 détaille les analogies entre SysML et Modelica et comment passer de l'un à l'autre. Des résultats sur la propulsion d'un navire sont présentés en section 4 avant de conclure.

2 Contexte des travaux

L'Ingénierie Système (IS) est une démarche méthodologique générale qui englobe l'ensemble des activités adéquates pour concevoir, faire évoluer et vérifier un système[1]. Cet article s'insère plus précisément dans le domaine de l'ingénierie système basée sur des modèles (MBSE) pour soutenir les différentes phases de développement d'un produit. Cette approche, qui a débutée dans les années 1950, regroupe aujourd'hui des organisations nationales (AFIS) et internationale (INCOSE), qui proposent des normes et des méthodes.

Le cycle en V est devenu le cycle de développement principal de l'IS. Ce cycle est séparé en deux branches. Dans la branche descendante sont réalisées la spécification et la conception du système, les exigences spécifiées sont validées au regard du niveau précédent et les activités de vérification et de validation système sont anticipées et planifiées en termes d'attendus. Une fois la phase de réalisation des composants élémentaires réalisée, la branche de droite du cycle constitue l'intégration, la vérification et la validation du système au regard du besoin initial [2].

La figure 1 place précisément l'article dans le contexte d'une conception basée sur un cycle en V. Le langage SysML est utilisé lors des phases de spécification du besoin et de conception architecturale, alors que le langage Modelica est utilisé lors des phases de conception architecturale et de conception détaillée. Bien que ces langages permettent de représenter des connaissances similaires, il n'y a pas de correspondance complète entre eux du fait de leurs finalités qui diffèrent. Cet article vise donc à détailler la méthode pour aider le concepteur durant la phase de conception architecturale en ciblant les analogies entre SysML et Modelica.

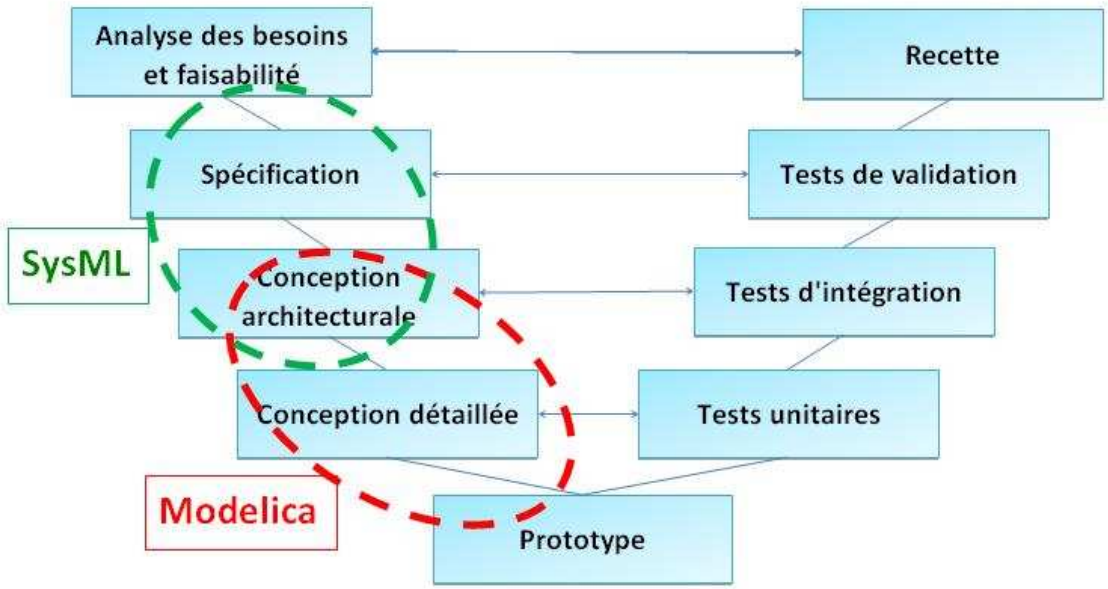

Figure 1: Phases de conception d'un système et cycle en V

1 http://www.afis.fr

2.1 SysML

SysML est un langage proposé par l'OMG[2] (Object Management Group) pour la modélisation de systèmes qui propose neuf diagrammes afin de couvrir au mieux toutes les facettes de modélisation. Il est dérivé d'UML[3] et partage avec ce dernier formalisme une partie de ses diagrammes (diagramme de cas d'utilisation, de séquence, d'états et de paquets). La sémantique de certains diagrammes a été modifiée pour mieux correspondre aux besoins de l'ingénierie système (diagramme d'activité, de définition de blocs et de blocs internes), alors que deux nouveaux diagrammes ont été définis (diagramme paramétrique et de besoins). Ces diagrammes sont présentés sur la figure 2 et sont regroupés selon trois types : comportement, fonctionnel et structure.

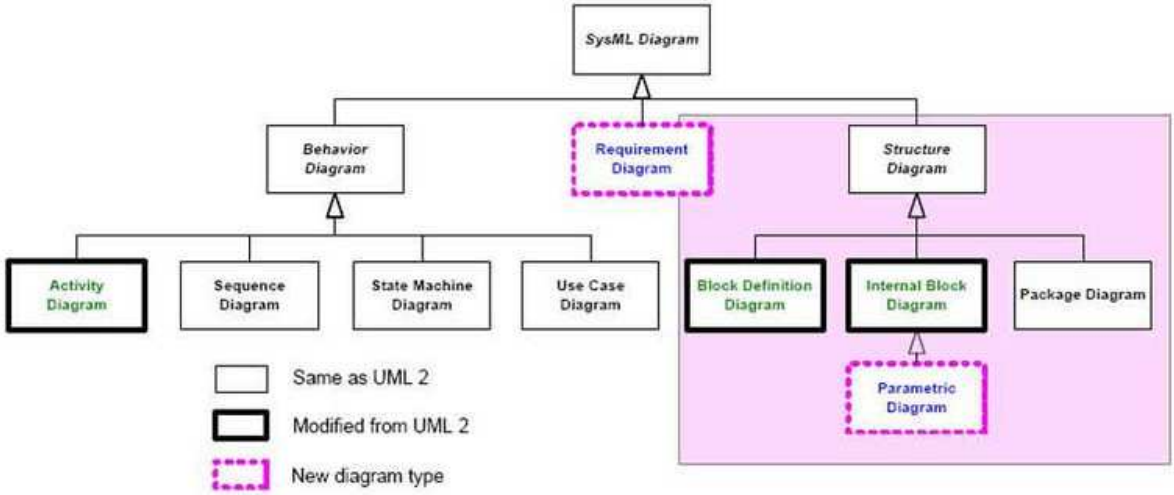

Figure 2: Organisation des diagrammes dans SysML 1.2 [4]

Dans cet article, nous nous intéressons plus particulièrement au diagramme de définition de blocs (BDD) et au diagramme de blocs internes (IBD) qui définissent principalement la structure du système étudié. Un bloc est l'élément principal de SysML. Il représente un système ou un sous-système. Il est défini par sa structure interne (contraintes, opérations…) et peut posséder des ports, précisant ses interfaces.

Le BDD est un diagramme qui permet de définir le système et sa hiérarchie. La sémantique utilisée est très proche du diagramme de classe d'UML et d'un organigramme technique. Un bloc peut représenter des sous-systèmes (composants) organiques ou fonctionnels. Des liaisons définissent les relations entre les blocs. L'entité spécification de flux permet de spécifier nature d'un flux pour un port.

La figure 3 présente un exemple de BDD avec un bloc propulsion qui contient un moteur et un réducteur. Une spécification du flux énergie de rotation est définie et des ports sont déclarés en sortie du moteur et en entrée du réducteur. La cardinalité au niveau des liaisons (1…*) indique que plusieurs moteurs peuvent être associé à plusieurs blocs propulsion et de même pour le réducteur. Cette description graphique permet donc de définir les composants d'un système, d'y spécifier leur entrées/sorties et de préciser leur lien de composition dans le système.

2 http://www.omg.org/
3 http://www.uml.org/

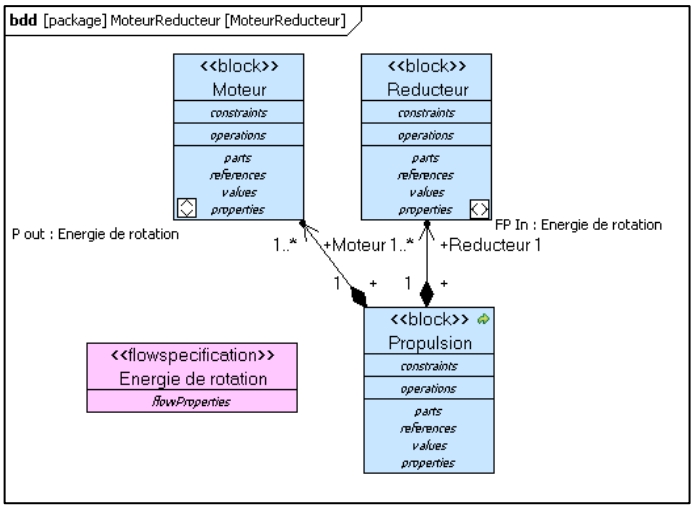

Figure 3: Exemple de Block Definition Diagram

L'IBD est un diagramme qui met en relief les liaisons entre les composants du système, le type de connexion et leur sens. Il ne précise pas le comportement du système mais les interactions (nature et sens) entre les composants au travers de la circulation des flux (énergie, matière, signal). Les parties représentent l'usage d'un bloc dans un contexte précis, en informatique il peut être associé à une instance de classe. Les ports peuvent être d'entrée, de sortie ou les deux ; les ports standards permettent le passage de différents types de flux alors que les autres ports se limitent à un seul. Les connecteurs caractérisent différents types de liaisons entre les parties. Dans l'exemple de la figure 4, les parties Moteur1 et Reducteur1 sont respectivement des instances des blocs Moteur et Reducteur. Ils sont reliés par leur port de type Energie de rotation et un connecteur C1 qui transporte cette Energie de rotation [5].

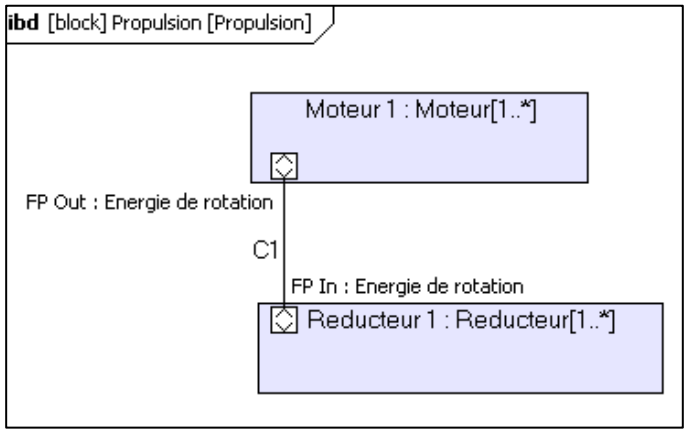

Figure 4: Exemple d'Internal Block Diagram

SysML est donc un outil puissant en termes de spécification de modèles avec une sémantique pour décrire les éléments du système. Néanmoins, il ne permet pas de faire des calculs et de la simulation de système dans un contexte donné contrairement à Modelica.

2.2 Modelica

Modelica est un langage textuel haut niveau pour la modélisation de système multiphysique. Les modèles qu'il permet de développer sont acausaux, basés sur des équations mathématiques et le langage est basé sur une approche orientée objet. Ces caractéristiques facilitent la réutilisation des connaissances décrites à travers ce langage.

L'utilisation de ce langage se fait tout d'abord en déclarant des modèles comme sur la figure 5 qui servira d'exemple pour ce paragraphe où un réducteur Gearbox est défini de la ligne 1 à la ligne 14. Ces modèles génériques vont alimenter une bibliothèque. Ils peuvent utiliser d'autres modèles prédéfinis en utilisant la fonction extends ligne 2. Les paramètres et les variables du modèle sont précisés respectivement aux lignes 3 et 4. Les équations précisent le comportement interne du système ; dans ce cas des composants plus basiques forment le système par connexion à la ligne 9.

```
1 model Gearbox "Realistic model of a gearbox (based on LossyGear)"
2   extends Modelica.Mechanics.Rotational.Interfaces.PartialTwoFlangesAndSupport;
3   parameter Real ratio(start=1) "transmission ratio (flange_a.phi/flange_b.phi)";
4   Modelica.SIunits.Angle phi_rel(start=0, stateSelect=stateSelect, nominal=1e-4);
5   equation
6      phi_rel = flange_b.phi - lossyGear.flange_b.phi;
7      w_rel = der(phi_rel);
8      a_rel = der(w_rel);
9      connect(flange_a, lossyGear.flange_a)
...
11   annotation (Icon(coordinateSystem(preserveAspectRatio=true, extent={{-100,-100},{100,100}}), graphics={
12               Rectangle(extent={{-100,10},{-60,-10}}, lineColor={0,0,0},
...
14 end Gearbox;
```

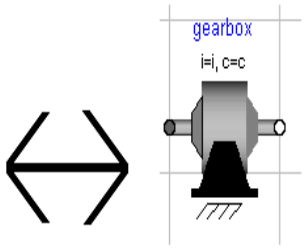

Figure 5:donne un exemple du modèle « inertia » de la bibliothèque standard de Modelica sous forme textuelle et graphique

Les modèles de la bibliothèque définis sont alors vus comme des composants ou des sous-systèmes du système. L'assemblage et le dimensionnement de ces composants permet la simulation du système. En pratique, le code Modelica est compilé pour générer un code de simulation exécutable.

Lorsqu'une interface adéquate est utilisée, les annotations de la ligne 11 sur la figure 5 permettent de représenter le code Modelica sous forme de composant graphique. L'image à droite de la figure 5 représente le graphique correspondant au modèle du réducteur sous l'interface SimForge.

3 Analogies entre SysML et Modelica

Ces deux langages de modélisation ont déjà fait l'objet de nombreuses études dont les travaux de C. Paredis et al [7] qui proposent une spécification pour la transformation bidirectionnelle de modèles de SysML à Modelica. Avant cela, des travaux sur la façon de lier plusieurs domaines grâce à SysML ont été réalisés [8]. Un exemple est donné entre EPLAN (logiciel pour la conception de structures géométriques 3D d'un produit) et Modelica qui spécifie les fonctions du système et de ses composants. Dans le même registre, ModelicaML est un langage de modélisation qui a été conçu pour générer du code Modelica à partir de modèles graphiques type UML ou SysML [9].

Cette section propose une synthèse des analogies existantes entre SysML et Modelica. Pour ces 2 langages, le système est vu comme un objet où sa structure est décrite. La table 1 propose des analogies entre les entités de chaque langage qui permettent cette description. Cette table reprend aussi les résultats des travaux cités précédemment.

Ces cinq définitions reprennent les entités principales pour la description d'un modèle préliminaire : le modèle avec ses caractéristiques internes et ses liaisons potentielles définies par les ports qu'il possède, les flux qui précisent les caractéristiques internes du connecteur (signal, matière, énergie), et enfin l'instance qui n'est rien d'autre qu'un modèle dimensionné et la connexion qui précise quelles instances sont reliées et par quel flux.

Déclaré dans	SysML	Modelica
Définition du modèle	« Block » (BDD)	« model » ou « block »
Définition du flux	« Flow spécification » (BDD)	« connector »
Définition du port	« Flow port » (BDD + IBD)	Variable de type connector
Définition de l'instance	« Part » (IBD)	Variable de type model
Définition de la connexion	« Connector » (IBD)	Equation « connect » entre 2 connecteurs

Table 1: Equivalence entre entités SysML et Modelica

Dans la figure 6, l'exemple du paquet moteur/réducteur est repris pour bien montrer ces analogies entre langages (ligne 1). Les données du tableau sont présentées et l'aspect bidirectionnel est signalé par les flèches à double sens. Le bloc Propulsion est notre système (Lignes 34 à 41). Il est composé du bloc moteur et du bloc réducteur (Lignes 8 à 32) dont leurs caractéristiques internes sont décrites dans Modelica (définition des variables, équations caractéristiques…). Les ports du moteur et du réducteur (Lignes 9 et 20) ayant la même spécification de flux (Lignes 3 à 6), ils peuvent être connectés. Dans le bloc propulsion, le moteur et le réducteur sont donc paramétrés (Ligne 35 et 36) et connectés (Ligne 40). Cet exemple représente un système propulsif simplifié mais suffisant pour illustrer les données de la table 1.

Ces analogies sont une base pour les développements présentés dans la partie 4. Il reste à définir les limites de ce « pont », car même si ces outils ont une approche système similaire, leurs fonctions et objectifs globaux sont différents.

4 Cas d'étude, exemple de la propulsion d'un navire

Les développements de la figure 7 sont basés sur un cas d'étude trouvé dans la littérature [10] qui fournit un modèle simplifié pour le système propulsif d'un cargo. Les résultats de SysML et de la simulation Modelica du paquet Propulsion d'un navire sont présentés. Le bloc propulsion contient un réservoir, un moteur diesel, un réducteur, une hélice et une coque. Des ports sont spécifiés pour décrire les flux qui peuvent potentiellement entrer ou sortir des blocs, ici le fuel, une énergie de rotation et une énergie de translation.

Pour le développement Modelica, l'exemple de la figure 7 est plus complet que précédemment mais le principe reste le même. Les connecteurs et modèles génériques sont déclarés des lignes 2 à 12. Le modèle de simulation contient un élément source qui permet de faire varier une variable au cours du temps (ligne 16) ; ici cette variable y varie linéairement de 7 à 17 sur un intervalle d'une seconde. y est ensuite affecté à la vitesse du navire (ligne 22), la simulation calcule donc les différentes variables du système en fonction de la vitesse qui varie de 7 à 17 mètre par seconde.

Un affichage des différentes variables est possible et les courbes de la vitesse du navire v, de la consommation mdot et du rendement R sont présentés en fonction du temps (figure 7). Le compilateur utilisé pour calculer ces résultats de simulation est OpenModelica.

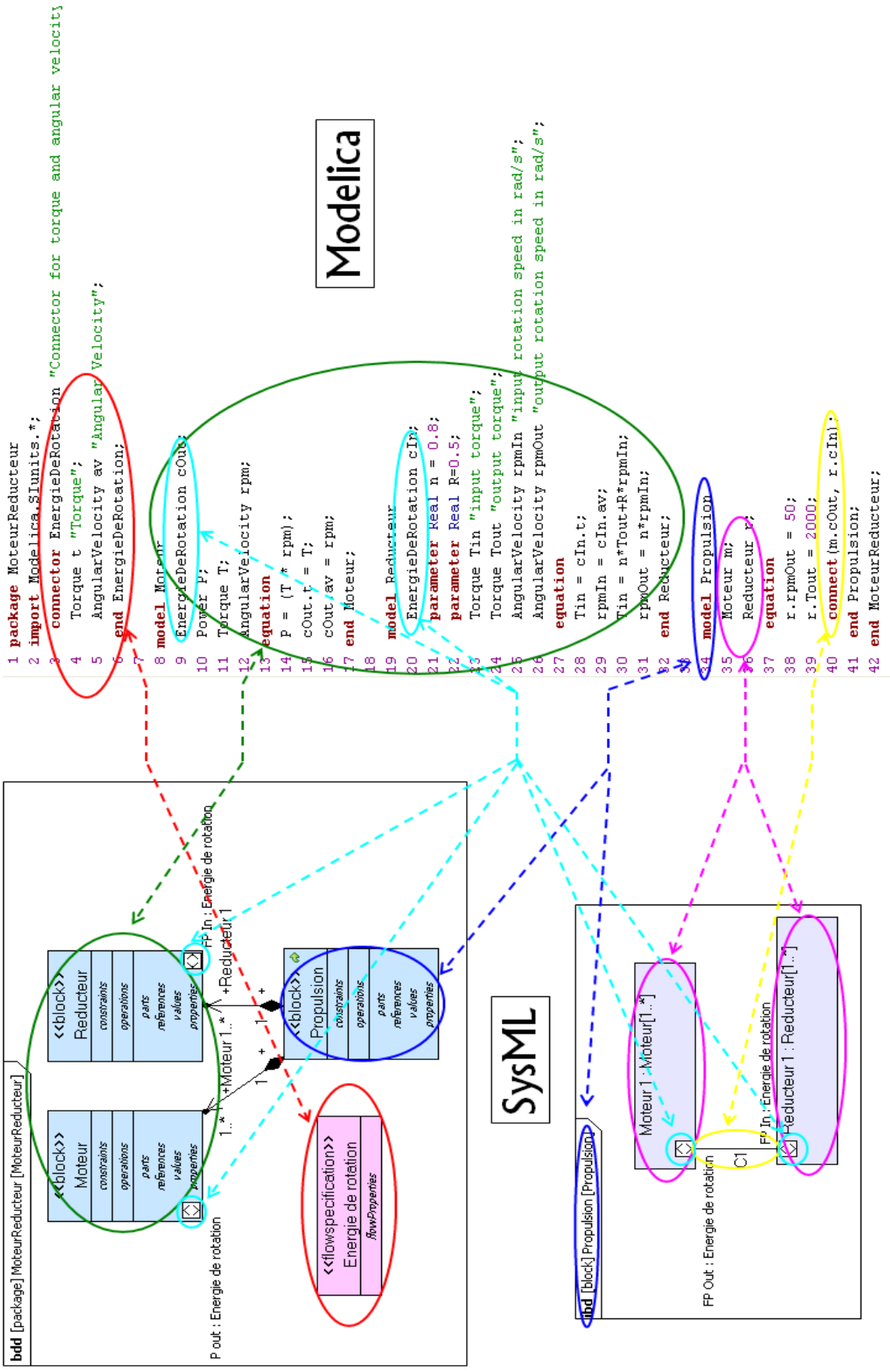

Figure 6: Exemple d'analogies entre SysML et Modelica

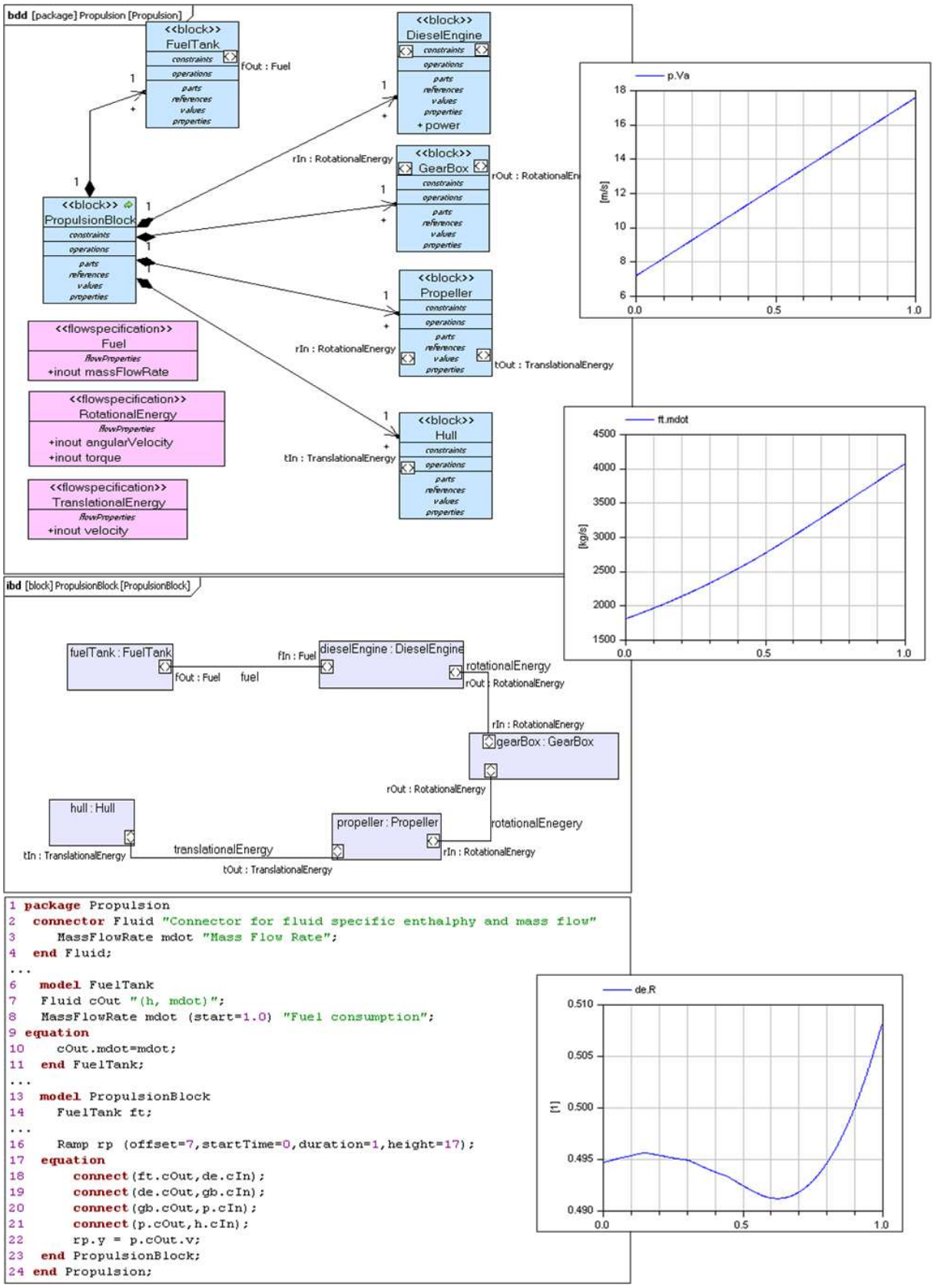

Figure 7:Modélisation SysML, Modelica et résultats de simulation OpenModelica de l'appareil propulsif d'un navire

5 Conclusion

Dans la phase de conception préliminaire d'un système complexe, cet article s'appuie sur une démarche d'ingénierie système basée sur les modèles pour justifier le choix de SysML et de Modelica. Le premier permet une modélisation graphique du système qui favorise donc la lisibilité et la compréhensibilité. Le deuxième permet une modélisation textuelle pour faire la simulation du comportement du système. Ces 2 langages décrivent la structure d'un système et de ses composants. Des analogies entre des entités de SysML à Modelica ont donc été présentées dans le but de réduire le temps de la conception préliminaire et de fiabiliser la transcription des connaissances. Un autre avantage de ces analogies est leur aspect bidirectionnel qui facilitera dans un deuxième temps le passage de Modelica à SysML lors des modifications dues aux phases de tests ou la réutilisation de modèles présents dans les bibliothèques Modelica. Comme le montre la figure 1, certaines fonctionnalités de SysML ne peuvent pas se traduire en Modelica et inversement car le domaine d'application des deux langages est différent comme en phase de spécification et phase de conception détaillée.

Il est important de rappeler que de nombreux travaux ont été réalisés sur ce sujet, et qu'il existe même un langage de notations graphiques qui permet de générer du code exécutable Modelica : ModelicaML [11]. Néanmoins Modelica ML est un profil UML qui modifie la sémantique d'UML pour représenter les concepts présents dans Modelica. Nous pensons qu'il est préférable de se baser sur l'ingénierie des modèles [12] pour mieux établir des correspondance entre Modelica et SysML tout en conservant leur sémantique initiale afin de ne pas dénaturer leur utilisation.

Références

[1] D. Scaravetti. « Formalisation préalable d'un problème de conception, pour l'aide à la décision en conception préliminaire », Thèse de doctorat, ENSAM, Décembre 2004.

[2] J.S. Gero, U. Kannengiesser. « The situated function–behaviour–structure framework », Key Centre of Design Computing and Cognition, University of Sydney, Sydney, 2006

[3] J.P. Calvez. « Spécification et conception des systèmes, une méthodologie », Masson, Paris, 1990.

[4] OMGSysML-v1.2, Spécification SysML, June 2010.

[5] S. Friedenthal, A. Moore, R. Steiner. « A Practical Guide to SysML: The Systems Modeling Language », Morgan Kaufmann; Elsevier Science, 2008.

[6] Modelica®. « A Unified Object-Oriented Language for Physical Systems Modeling » Language Specification Version 3.2, March 24, 2010.

[7] C. Paredis, Y. Bernard, R.M. Burkhart, H.P. de Koning, S. Friedenthal, P. Fritzson, N. F. Rouquette, W. Schamai. « An Overview of the SysML-Modelica Transformation Specification », 2010 INCOSE International Symposium, July 2010.

[8] A. Shah, A. Kerzhner, D. Schaefer, and C. Paredis. « Multi-View Modeling to Support Embedded Systems Engineering in SysML », Lecture Notes in Computer Science, Springer, 2009.

[9] W. Schamai, P. Fritzson, C. Paredis, A. Pop. « Towards Unified System Modeling and Simulation with ModelicaML: Modeling of Executable Behavior Using Graphical Notations », Modelica Conference, 2009.

[10] B. Cuneo, T. McKenney, M. Parker. « Design Optimization Study on a Containership Propulsion System », Final report, April 2010.

[11] W. Schamai. « Modelica Modeling Language (ModelicaML), a UML Profile for Modelica », Linköping University, EADS Innovation Works, 2009.

[12] J. Bézivin, F. Jouault, P. Rosenthal, and P. Valduriez. « Modeling in the Large and Modeling in the Small », University of Nantes, Springer, 2005